**DPU Update Script Design doc 1.1**

Contents

[1. Description 1](#_Toc197681845)

[2. Design 1](#_Toc197681846)

[2.1. Architect 1](#_Toc197681847)

[2.2. Command parameters 2](#_Toc197681848)

[2.3. Normal output 2](#_Toc197681849)

[2.4. Error List 3](#_Toc197681850)

[2.5. Log and Debug 4](#_Toc197681851)

[2.6. Limitations 4](#_Toc197681852)

[3. Workflow 5](#_Toc197681853)

[3.1. BMC firmware update 5](#_Toc197681854)

[3.2. CEC Firmware update 5](#_Toc197681855)

[3.3. ATF/UEFI firmware update 6](#_Toc197681856)

[3.4. BUNDLE firmware update 8](#_Toc197681857)

# Description

DPU update script (OobUpdate.sh) is a program for updating various component firmware of Bluefield DPU, like BMC, CEC and BIOS(ATF&UEFI). It works from out of band, using Redfish API exposed by BMC of DPU. The script can work from any controller host (Linux), which has available connection to the DPU BMC system.

# This design document corresponds to version 25.04-2.1 of the script.

# Design

## Architect

The DPU Update Script works as following chart:

ARM OS

BMC

ATF/UEFI

DPU

Redfish(Restful) API

DPU Update  
Script

Controller Host

* DPU Update Script can run from any machine (Controller Host), which can establish connection to DPU BMC’s OOB interface.
* The script connects and manages the DPU through Redfish (Restful) API exposed by DPU BMC.
* The script is mainly implemented by Python3.

## Command parameters

OobUpdate.py [-h] [-U <username>] [-P <password>]

[-S <ssh\_username>] [-K <ssh\_password>] [-F <firmware\_file>]

[-T <module>] [--with-config] [-H <bmc\_ip>] [-C]

[-o <output\_log\_file>] [-p <bmc\_port>] [--config <config\_file>]

[-s <oem\_fru>] [-v] [--skip\_same\_version] [-d]

[-L <path>] [--task-id <task\_id>] [--lfwp]

options:

-h, --help show this help message and exit

-U <username> Username of BMC

-P <password> Password of BMC

-S <ssh\_username> Username of BMC SSH access

-K <ssh\_password> SSH password of BMC

-F <firmware\_file> Firmware file path (absolute/relative)

-T <module> The module to be updated: BMC|CEC|BIOS|FRU|CONFIG|BUNDLE

--with-config Update the configuration image file during the BUNDLE update process.

-H <bmc\_ip> IP/Host of BMC

-C Reset to factory configuration (Only used for BMC|BIOS)

-o <output\_log\_file>, --output <output\_log\_file> Output log file

-p <bmc\_port>, --port <bmc\_port> Port of BMC (443 by default).

--config <config\_file> Configuration file

-s <oem\_fru> FRU data in the format "Section:Key=Value"

-v, --version Show the version of this scripts

--skip\_same\_version Do not upgrade, if upgrade version is the same as current running version

-d, --debug Show more debug info

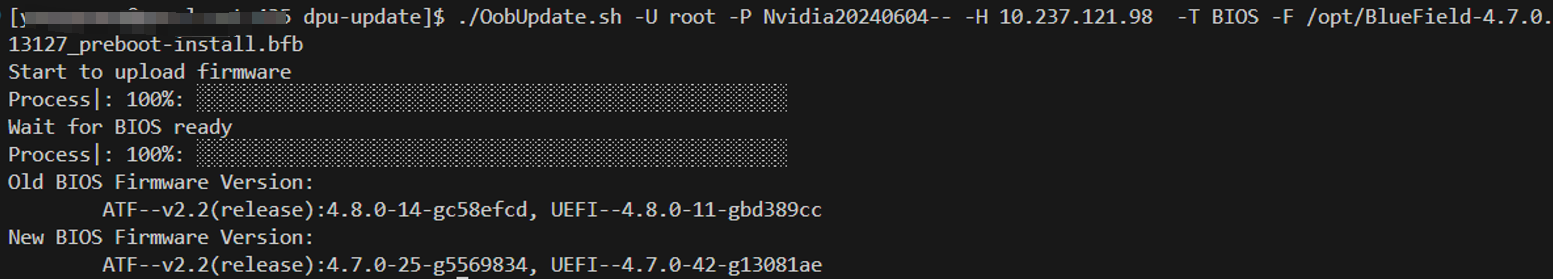
-L <path> Linux path to save the cfg file

--task-id <task\_id> Unique identifier for the task

--lfwp Live Firmware Update patch. Works only with BUNDLE module.

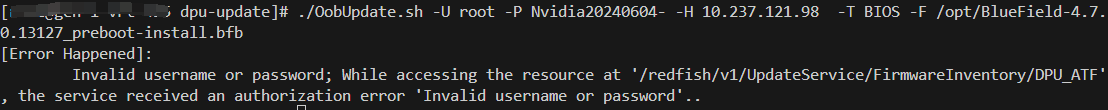
## Normal output

In normal successful case, the script will only have a few messages show in standard output. Like following:



## Error List

While this script is running, errors may be encountered. Related error message will show in error output.

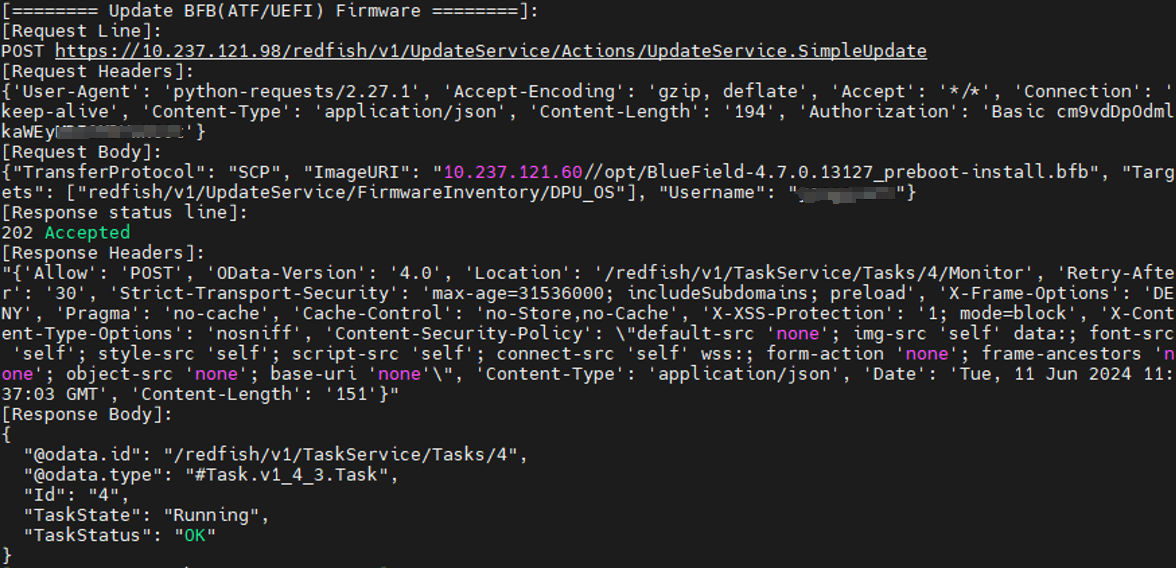


Following is error list the script may encounter:

|  |  |  |
| --- | --- | --- |
| Index | Error Name | Error Message |
| 1 | ARG\_FOR\_UPDATE\_NOT\_GIVEN | BMC IP/Username/Password, Firmware file path and Module are needed to do firmware update |
| 2 | FILE\_NOT\_ACCESSIBLE | File is not accessible |
| 3 | FW\_FILE\_NOT\_MATCH\_MODULE | Given firmware file is NOT for the Module to update |
| 4 | BMC\_CONNECTION\_FAIL | Failed to establish connection to BMC. Please check the BMC IP and port |
| 5 | BMC\_CONNECTION\_RESET | Connection to BMC being reset by remove |
| 6 | ACCOUNT\_LOCKED | Account has been locked |
| 7 | INVALID\_USERNAME\_OR\_PASSWORD | Invalid username or password |
| 8 | ANOTHER\_UPDATE\_IS\_IN\_PROGRESS | Another update is in progress; Please try to update the firmware later |
| 9 | UNSUPPORTED\_MODULE | Unsupported updating module |
| 10 | BAD\_RESPONSE\_FORMAT | Bad response format |
| 11 | INVALID\_STATUS\_CODE | Invalid response status code |
| 12 | FAILED\_TO\_GET\_LOCAL\_KEY | Failed to get local SSH Key |
| 13 | FAILED\_TO\_ENABLE\_BMC\_RSHIM | Failed to enable BMC rshim; Please make sure rshim on Host side is disabled |
| 14 | NOT\_SUPPORT\_CEC\_RESTART | CEC restart redfish API is not supported in this version; Please use power cycle of the whole system instead |
| 15 | BMC\_BACKGROUND\_BUSY | BMC is busy on background operation; Please try to update the firmware later |
| 16 | PUBLIC\_KEY\_NOT\_EXCHANGED | Public key was not exchanged with BMC successfully |
| 17 | BIOS\_FACTORY\_RESET\_FAIL |  |

## Log and Debug

There are some additional logs recording details of each restful request & response. Those logs can be used in debugging issues.



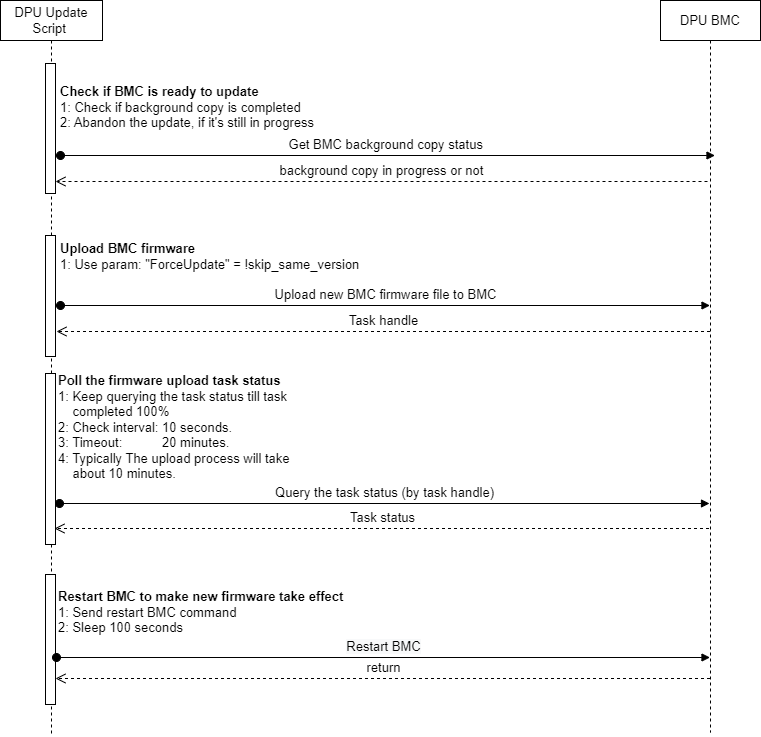
* If command has argument -d (--debug) , those additional logs will be shown in output
* If command has argument -o (--output\_log\_file), those additional logs will be written into the log file.

## Limitations

1. The script can only run on Linux host, with python3 installed
2. The rshim on Host side (in which DPU plugged) need to be disabled, if want to update BIOS firmware of DPU or use BUNDLE module
3. The target DPU BMC version should be >= 24.04
   * Some redfish API may not be supported for earlier BMC versions.
   * That also means the script can only support >= Bluefield 3
   * The DPU BMC needs to be upgraded to >= 24.04 by other means, before using this script.
4. The target DPU BMC version should be >= 25.04
   * To support Live Firmware Patch (LFWP)
5. The default user/password of DPU BMC should be updated in advance.
6. For one DPU system, there can only be one update task at the one time. The later coming task will get “Another update is in progress" error.
7. In NIC mode, the BMC's Redfish chassis schema contains only limited information about BlueField (like versions of different components). This is because, in this mode, the OS is not available to supply the necessary information to the BMC.

# Workflow

# BMC firmware update



# CEC Firmware update

The CEC firmware update workflow is similar as BMC firmware update. Except for following:

1. The upload process will take less than 4 minutes
2. Will check task status every 2 seconds
3. Will wait 120 seconds, after CEC restart.

# ATF/UEFI firmware update

A white paper with black text

Description automatically generated

# BUNDLE firmware update

The bf-fwbundle BFB update process uses the Redfish **SimpleUpdate** API to transfer firmware bundles to the BMC. Key stages:

1. **OobUpdate** validates arguments and confirms BMC SSH credentials are provided.
2. Creates bf.cfg configuration file and appends to the original bf-fwbundle BFB:

BMC\_USER=<Redfish\_username>

BMC\_PASSWORD=<Redfish\_password>

BMC\_SSH\_USER=<SSH\_username>

BMC\_SSH\_PASSWORD=<SSH\_password>

BMC\_REBOOT="yes"

CEC\_REBOOT="yes"

1. Checks for config-image.bfb presence in the bf-fwbundle to determine if **ResetBios** is required post-update.

Note: config-image.bfb will not be updated by default. To enable config-image.bfb update use ‘--with-config’ parameter.

1. Parses the info json file coming with the bf-fwbundle to extract component versions from the bundle.
2. BFB update sequence:
   1. Enable RSHIM on the BMC via Redfish API
   2. Initiates **SimpleUpdate** task to transfer BFB via RSHIM
      1. Simple Update sends the bf-fwbundle BFB to the DPU via RSHIM interface and finishes once the BFB is fully uploaded. During this time the DPU resets and starts running the BFB installation environment.
      2. BFB installation environment is running on the DPU. Update sequence:
         1. ATF/UEFI firmware capsule
         2. BMC Firmware (if not the same)
         3. BMC restart to apply the new version (disconnects BMC console)
         4. CEC Firmware (if not the same)
         5. CEC restart to apply the new version
         6. DPU Golden Image
         7. NIC Firmware Golden Image (if exists)
         8. config-image.bfb
         9. NIC Firmware
         10. NIC Firmware reset using mlxfwreset to apply the new version (if supported in current state)
         11. Reset DPU to finish the installation
         12. New ATF/UEFI activated
         13. Load DPU OS
3. Waits for the DPU readiness
4. DPU reset to apply the new configuration from the config-image.bfb
5. Collects new versions of all the running components
6. Generates version comparison table

